Branch-and-Terminate: a combinatorial optimization algorithm for protein design

D Benjamin Gordon[1] and Stephen L Mayo[2]*

Background: Several deterministic and stochastic combinatorial optimization algorithms have been applied to computational protein design and homology modeling. As structural targets increase in size, however, it has become necessary to find more powerful methods to address the increased combinatorial complexity.

Results: We present a new deterministic combinatorial search algorithm called 'Branch-and-Terminate' (B&T), which is derived from the Branch-and-Bound search method. The B&T approach is based on the construction of an efficient but very restrictive bounding expression, which is used for the search of a combinatorial tree representing the protein system. The bounding expression is used both to determine the optimal organization of the tree and to perform a highly effective pruning procedure named 'termination'. For some calculations, the B&T method rivals the current deterministic standard, dead-end elimination (DEE), sometimes finding the solution up to 21 times faster. A more significant feature of the B&T algorithm is that it can provide an efficient way to complete the optimization of problems that have been partially reduced by a DEE algorithm.

Conclusions: The B&T algorithm is an effective optimization algorithm when used alone. Moreover, it can increase the problem size limit of amino acid sidechain placement calculations, such as protein design, by completing DEE optimizations that reach a point at which the DEE criteria become inefficient. Together the two algorithms make it possible to find solutions to problems that are intractable by either algorithm alone.

Addresses: [1]Division of Chemistry and Chemical Engineering, California Institute of Technology, Pasadena, California 91125, USA and [2]Howard Hughes Medical Institute and Division of Biology, 147-75, California Institute of Technology, Pasadena, California 91125, USA.

*Corresponding author.
E-mail: steve@mayo.caltech.edu

Key words: Branch-and-Bound, dead-end elimination, global energy minimum, protein design, rotamers

Received: 8 February 1999
Revisions requested: 23 April 1999
Revisions received: 14 May 1999
Accepted: 18 May 1999

Published: 26 August 1999

Structure September 1999, 7:1089–1098
http://biomednet.com/elecref/0969212600701089

0969-2126/99/$ – see front matter

Introduction

Significant advances in protein design [1,2] and protein sidechain homology modeling [3] have arisen from the application of optimization algorithms and specialized potentials to the sidechain placement problem. In these calculations, one searches for the set of sidechain conformations that produce the global minimum energy conformation (GMEC) for the given protein backbone. The energies of sidechain interactions are evaluated using empirically based energy potentials and, to reduce the complexity of the calculation, the set of possible sidechain orientations is discretized into statistically representative conformations called rotamers [4,5].

The search for the optimal selection of sidechain rotamers for a specified protein fold is necessarily a combinatorial optimization problem; an exhaustive search through all combinations is intractable. As such, the problem has been approached by several different methods, including Monte Carlo [6,7] and simulated annealing [8], mean-field [9,10], and dead-end elimination (DEE) [11–14]. In particular, DEE methods have emerged as powerful tools for more difficult protein design calculations, in which the optimal sidechains are selected from rotamers of many different amino acids [1,2].

There are optimizations, however, for which DEE algorithms are not sufficient, due to either the nature of their energy distributions or their sheer size. For example, the optimization of long hydrophilic sidechains on β sheets is typically composed of large numbers of rotamers with interaction energies that are very small in magnitude. DEE is able to reduce the combinatorial size of the problem significantly at the outset, but, soon after, elimination becomes inefficient, relying entirely on computationally expensive DEE doubles calculations [12,14]. This behavior is also observed in the later stages of very large calculations, when after several rounds of unification [15] further eliminations become difficult and the number of super-rotamers at super-residue positions becomes very large. To complete such calculations, a technique consisting of exhaustive combinatorial build-up aided by DEE has been described [3]. However, as the effectiveness of the elimination criteria is poor in these cases, it is advantageous to construct a method that is not dependent on them.

To address these difficult optimization problems, we have developed an enhanced version of a Branch-and-Bound (B&B) algorithm [16] that we have dubbed 'Branch-and-Terminate' (B&T). B&B algorithms comprise a subclass

of backtrack algorithms that utilize information about costs (or energies) of complete and partial solutions. Backtrack algorithms are commonly used in atomic-level simulations to construct self-avoiding chains, and they have been used in protein design to engineer metal-binding sites into proteins [17].

B&B algorithms are commonly applied to theoretical combinatorial and scheduling problems, and, more recently, to combinatorial problems of structural biology ranging from sequence alignment [18] and structural comparison [19], to macromolecular packing [20], ligand design [21] and, recently, protein tertiary structure prediction [22]. Regarding the study of protein sidechains, Samudrala and Moult [23] have described a graph-theoretic approach to the closely related problem of comparative modeling, in which they represent the search as a clique-finding problem that they solve using a B&B algorithm. In addition, Leach and Lemon [24] have used a B&B algorithm (called 'A*') to explore the conformational energy surface of protein sidechains.

It is straightforward to formulate the sidechain optimization problem for direct optimization by a B&B algorithm. All that is necessary is to describe the problem as a search of a combinatorial tree where one searches for the single path through the branches that corresponds to the GMEC set of rotamers. The B&B algorithm is effective because it simultaneously prunes the tree while searching; each branch is tested with a quantitative bounding expression before being searched.

In implementing a B&B algorithm for sidechain selection, we have incorporated some novel algorithmic techniques that increase the optimization speed dramatically. First, we describe a bounding function that maximizes the efficiency of pruning for problems in which the total energy can be decomposed into interactions between pairs of rotamers. We also describe a process we call 'termination', in which we use the bounding function to deterministically remove rotamers at all amino acid positions, thereby reducing the overall size of the tree before searching. Termination is additionally effective when performed at every level of recursion of the search, sometimes increasing the overall speed of the optimization by an order of magnitude. Last, we demonstrate how the energetic information produced by the termination process can be used to determine the optimal search order for the remainder of the tree. Because termination effectively replaces the usual bounding process, the resulting breadth-first algorithm is called 'Branch-and-Terminate'. We also describe a variation of the B&T method that can rapidly find approximate solutions close to the GMEC.

The description of the B&T algorithm that follows is tailored for rotamer selection, but the algorithm can in fact be generalized to any combinatorial optimization problem in which all the interaction energies are pairwise and precomputable. The bounding expression we describe is similarly general.

Although the B&T algorithm can be used by itself, greater benefit can often be obtained by using it in concert with a DEE algorithm. Together, the algorithms can solve optimization problems much more quickly than either can accomplish alone. This may make it possible to quickly find the GMEC for protein design problems that were previously insoluble by either algorithm.

Results

Branch-and-Bound

When a combinatorial tree is used to describe the sidechain optimization problem, the root of the tree is placed at the top and branches extend downwards. Each level of depth of the tree corresponds to an amino acid position, and each node represents a particular rotamer choice at that position. Thus a path that extends all the way from the tree root through all levels of branches to a leaf describes a complete rotamer sequence. The problem, then, is to search for the path corresponding to the sequence with the lowest energy.

A partial path from the root describes a rotamer sequence that is incompletely specified. Alternatively, the path can be interpreted physically as specifying a unique composite rotamer, or 'super-rotamer', that occupies a subset of the amino acid positions. Extending the path deeper into the tree corresponds to appending additional rotamers to the super-rotamer, which can be repeated until all positions are specified. According to this interpretation, a full search of the tree would entail the construction of all possible super-rotamers to completion.

It is often possible, however, to determine that a particular partially specified super-rotamer is not part of the GMEC. In such a case, it is unnecessary to explore any combinations that would result from building up the super-rotamer further. Applied recursively, such observations prune subtrees from nodes throughout the tree, thereby enabling an exhaustive search without complete enumeration of all possible super-rotamers.

The pruning determination is accomplished by comparing a lower energy bound for the partially specified rotamer sequence to a known reference energy. Given a reference energy of any plausible sequence, it must be true that the energy of the GMEC is less than or equal to the energy of any plausible sequence:

$$E_{GMEC} \leq E_{reference} \quad (1)$$

One may therefore deduce that the global minimum does not contain a particular super-rotamer upon observing that

the energy $E_{super,best}$ of the sequence resulting from optimal completion of the candidate super-rotamer is greater than the reference energy:

$$E_{super,best} > E_{reference} \quad (2)$$

Finding the optimal completing sequence, however, can be as difficult as the original problem, so we instead construct an expression for a lower energy bound, $E_{super,bound}$. The expression is constructed to compute an inexpensive lower energy bound based on the partially specified sequence, as well as on the rotamers that are available at the unspecified positions. By definition, the bound must satisfy the inequality

$$E_{super,best} \geq E_{super,bound} \quad (3)$$

With this quantity in hand, we may prune any subtree for which we observe that the lower bound is greater than the reference energy:

$$E_{super,bound} > E_{reference} \quad (4)$$

This is the bounding criterion. The B&B algorithm consists of an exhaustive traversal of the combinatorial tree, applying this criterion to each node as it is encountered. Whenever the search produces a complete path with an energy lower than the current reference energy, the reference energy is updated. In this way, the effectiveness of the bounding criterion is increased over the course of the optimization. Moreover, upon completion of the search, the reference energy is the global minimum energy. The corresponding sequence is also stored during each update, which produces the corresponding GMEC.

Bounding expression

The successful implementation of a B&B type of algorithm depends largely on the construction of the bounding expression. A bounding expression that is very stringent will produce lower bounds that are high in energy, and therefore will result in more subtrees that can be pruned by the bounding criterion. The size of the resulting tree will be smaller than one pruned by a less stringent expression, and the search will be faster. It is therefore important to design the bounding expression to most fully utilize the sequence information available.

On the other hand, stringency is obtained at the cost of time. A maximally stringent bound might prune all subtrees except for the one containing the global minimum, but it would take an impractical amount of time to compute. It is therefore also necessary to temper stringency with speed considerations in order to obtain a bounding expression that is properly balanced for efficient searching.

We describe the construction of such a bounding expression in the Materials and methods section. Given a partially constructed super-rotamer and the available rotamers at the remaining positions, the approach is to utilize the corresponding energetic information as fully as possible while keeping the computational order of the bounding expression constant. The result is a novel, highly effective bounding expression that provides the basis for the remaining B&T techniques.

The form of the resulting expression has an additional advantage: it isolates those parts of the expression that are identical for rotamers on the same level of a subtree. Thus it is possible to further increase the efficiency of the search by precomputing these shared quantities as each group of nodes is encountered, rather than redundantly evaluating the entire bounding expression for every unique node. This method is described in the Materials and methods section.

Termination

The enhancements of the B&T algorithm relative to the B&B method are based on a process called ‘termination’. Because all the pairwise interactions are precomputable, the organization of the combinatorial tree is arbitrary (i.e. there is no specific order in which different amino acid positions must be assigned to different levels of the tree). The organization of the tree can, however, have a significant influence on the speed of the calculation. For example, a greater reduction in the size of the search is derived from pruning a branch at the root of the tree rather than pruning a branch closer to the leaves. Placing a branch at the leaves that would be pruned if placed at the root would be inefficient because the same pruning step would necessarily be repeated for every leaf.

In fact, it commonly occurs that all amino acid positions have some rotamers that could be pruned if placed at the root of the tree. To circumvent the potential loss of efficiency, we implement a preprocessing procedure before determining the tree organization. This procedure consists of temporarily considering each amino acid position to be at the root level and checking if any of its rotamers can be immediately pruned. All rotamers pruned from root positions may be completely discarded for the remainder of the optimization, and are dubbed ‘terminated’ to reflect this fact. The result is an overall reduction of the tree size prior to searching, making the optimization faster.

The selection of the word ‘terminate’ is intended to be contrasted with ‘eliminate’, which is used to describe rotamers that are analogously discarded by using the DEE criterion. Indeed, many of the same rotamers are

discarded. As with DEE, termination may be performed iteratively until no further rotamers are terminated. Iterative termination is executed as the preprocessing step before search of the tree.

Recursive termination

Although termination serves as an effective preprocessing step, the hallmark of the B&T algorithm is that termination is employed at every level of recursion. At any point of the search, the rotamers defined at levels above the level of the current amino acid position may be considered a root comprised of a single, partially specified super-rotamer. Termination, then, consists of temporarily considering each of the rotamers at all the remaining positions as candidates for the next appendage of the super-rotamer and applying the bounding criterion to each one. All rotamers terminated this way may be discarded from the optimization of the subtree with this partially specified super-rotamer root.

In contrast, the recursive step in a B&B search consists of the application of the bounding criterion to the rotamers at only one amino acid position. The benefits of the extra reductions in the sizes of subtrees far outweigh the costs of calculation of extra bounds for termination. The resulting increase in efficiency makes the B&T search significantly faster than a similarly constructed B&B search.

We have observed that it is not necessary to perform iterative termination at every level of recursion, unlike termination preprocessing. A single iteration per branch generally yields the best performance.

Search order

When traversing the combinatorial tree, it is necessary to determine the order in which to explore rotamers at each position and the sequence in which to explore the different positions. For both cases, we utilize the bounding energies calculated for each rotamer during termination.

We have observed an empirical correlation between low bounding energy and membership in the GMEC; therefore, the rotamers at each position are searched in order of increasing bounding energy. Conducting the search in this way increases the chance that solutions close to the GMEC are found quickly, thereby providing stringent reference energies early in the calculation.

With respect to the ordering of the different positions, we construct a heuristic based on both the termination bounding energies and the size of the rotamer lists. In a conventional tree search, the positions should be organized in order of increasing number of rotamers per position in order to minimize the total number of nodes in the tree. However, in a B&T search, there are other organization schemes that favor high-level pruning by termination that reduce the tree size more significantly. We use the bounding energy of the top-ranked (lowest bounding energy) rotamer at each position to indicate which positions are likely to restrict the rest of the system, and consequently favor high-level termination if placed at the super-rotamer root. Because the minimum operators at a node are applied over a set including the subset corresponding to the subtree nodes, bounding energies of subtree nodes must be higher than or equal to their parent nodes. Therefore, placing positions with high lowest-energies at the top of the tree promotes high bounding energies for their descendents. Because the rotamer lists of a subtree can be significantly different from those of its parent, residue ordering is performed at every level of recursion depth.

We have observed that an optimal ordering can be obtained by combining energetic and list-size sorting criteria using the following heuristic. Positions are sorted in descending order according to a rank index, as computed by the expression

$$\text{Rank index} = (1-f)\frac{1}{1+\ln N} + f\frac{E_{top} - E_{top,min}}{E_{top,max} - E_{top,min}} \qquad (5)$$

where N is the number of rotamers at the position, E_{top} is the bounding energy of the top-ranked rotamer of that position, and $E_{top,min}$ and $E_{top,max}$ are the minimum and maximum top-ranked bounding energies of all positions, respectively. The expression $1/(1 + \ln N)$ is constructed to produce an attenuated weighting inversely proportional to the number of rotamers that evaluates to unity when $N = 1$. The quantity f is selected to control the relative weighting of the two criteria. A value of zero for f corresponds to sort based entirely on the number of residues per position, and a value of one produces a ranking based entirely on bounding energies.

Approximate algorithm

A solution that is very close to the GMEC sequence can be found very rapidly by using an approximate variation of the B&T method. Approximate calculations are particularly useful for providing a fast way to obtain low reference energies for exact B&T optimizations. Moreover, the approximate calculation is often sufficient to produce the GMEC energy.

The approximation is based on the observation that the GMEC rotamers are often among those with the lowest termination bounding energies according to the bounding expression (Equation 21 in Materials and methods section). This indicates that the bounding expression has predictive properties. To rapidly find an approximate solution, the ranked rotamer lists are arbitrarily truncated after the preprocessing termination step and

the B&T search is conducted on the abbreviated set of rotamers.

A more reliable solution can be found by repeating the approximate optimization with more lenient truncation, using the solution from the preceding run for the initial reference energy.

DEE preprocessing

Perhaps the most practical use of the B&T algorithm is to complement DEE when dealing with optimization problems that are too difficult to solve using either algorithm alone. In such cases, the algorithms are used in succession. DEE is used to eliminate rotamers and to perform unification until the optimization reaches iterations that are inefficient. Inefficiency typically occurs after several unifications when the total number of rotamers and unified super-rotamers becomes very large (>5000) and very few eliminations result even from lengthy Goldstein doubles calculations. At this stage, the DEE optimization is aborted, and the state information is transferred to a B&T implementation. Rotamer lists and energy tables are transferred directly, including references to unified super-rotamers, which are transparently represented as ordinary rotamers in the B&T algorithm.

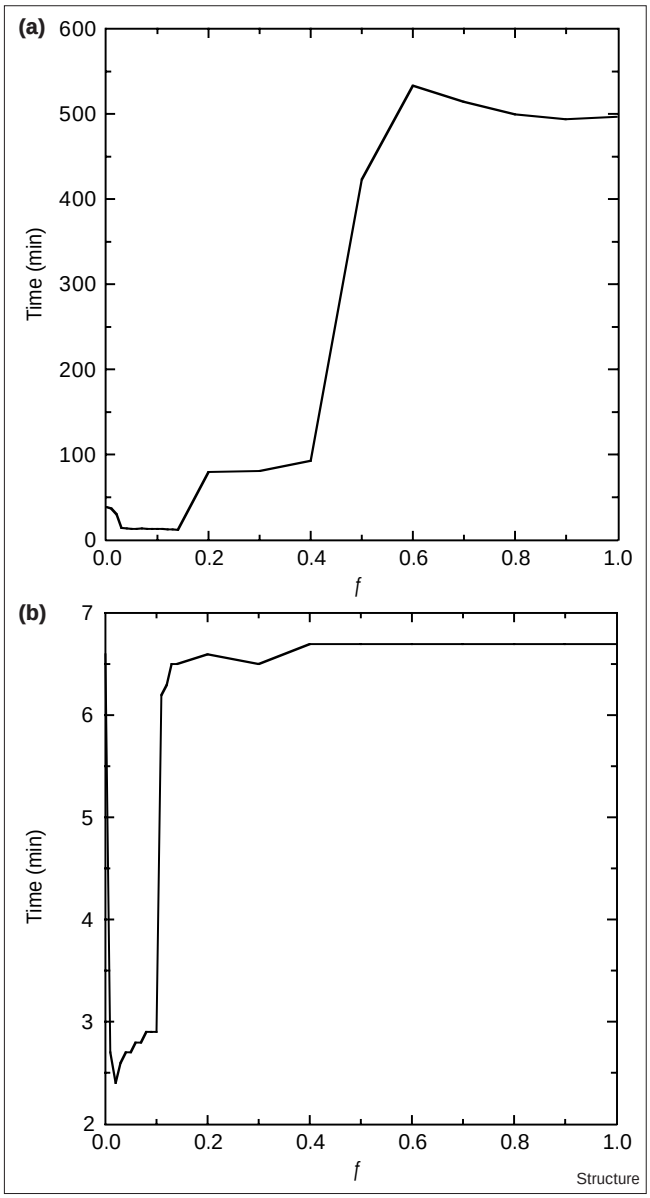
An additional performance improvement is obtained by also passing the list of dead-ending pairs (DEPs). DEPs are pairs of rotamers (or super-rotamers) the members of which cannot simultaneously exist in the GMEC. These pairs may therefore be safely omitted from the minimum operators in Equation 21 (see Materials and methods section).

Benchmarks

To assess the generality of the B&T approach, different incarnations of the algorithm were applied to benchmark problems representing different structural classes, as described in the Materials and methods section. Optimization times were heavily dependent on the sorting heuristic, as shown in Figure 1. The performance improvement, as measured by dividing the total optimization times, ranged from a factor of three for the case of the β sheet to a factor of over 40 for the 'mixed' case. Remarkably, very similar values of the sorting factor f produced the fastest optimization times for all structural classes. Initially, values at intervals of 0.1 were tested, but as all benchmark cases exhibited minima near $f = 0.1$, values at intervals of 0.01 were sampled near this value. At this level of refinement, the different cases had different optimal sorting factor values, but a value of $f = 0.08$ was close to optimal for all of them. We also observe that optimizations with the fastest times had the fewest nodes in their pruned combinatorial trees.

The total calculation times for the benchmarks using a sorting factor of 0.08 are competitive compared to times of a highly optimized DEE algorithm, and are significantly faster than the optimized B&B search (Table 1). For the β-sheet surface and the small core-boundary calculations, the B&T method is approximately 20 times faster than DEE. For the mixed case, it is nearly eight times faster. For the α-helical case, however, the B&T method is more than two times slower. This is likely to be a reflection of the linear topological arrangement of the system, in which it is difficult to select positions to place at the tree root

Figure 1



Total optimization time versus value of sorting method for (a) the mixed structural type and (b) the β-sheet surface benchmark cases. Sorting is determined by the value of the factor f in Equation 5. The cases exhibit different dependencies on the value of the sorting factor, but both have minima in the vicinity of $f = 0.08$. This trend is observed for all cases (not shown).

Table 1

Benchmark times.

	Benchmark cases				
	Small core boundary*	α Surface	β Surface	Mixture	Core boundary
Total times (min)					
DEE†	177.4	2.2	40.5	101.6	1154.0
B&B‡	70.7	294.8	44.4	544.9	> 30,000§
B&T#	8.4	6.1	2.1	13.0	745.8
B&T component times (min)					
Preprocessing	0.1	0.1	0.1	0.4	0.6
Search	8.3	6.0	2.0	12.4	744.8
Approximation¶				0.2	0.4
B&T total nodes	3829	1697	1546	845	34,634

*Refers to the benchmark comprised of a small set of core and boundary positions. †DEE was performed using the speed enhancements described in [13] and [14]. ‡The B&B algorithm uses the novel bounding expression and includes termination preprocessing. §For the difficult core boundary case, the incomplete B&B optimization was aborted after 30,000 min. #Total B&T time is computed as the sum of the approximation, preprocessing and search times. ¶An approximate B&T algorithm was used to obtain initial bounds for the mixture and difficult core boundary cases. These calculations used only the top thirty rotamers at each position according to their bounding energy.

that both restrict large parts of the system and are themselves restricted.

The approximate form of the algorithm proved to be exceptionally effective. For the four cases above, B&T calculations that used only the 30 top-ranked rotamers at each position all took less than 15 s and produced the correct GMEC solutions. For the more difficult core-boundary case, the calculation took 5 min, and also produced the correct GMEC solution. For this case, a more aggressive calculation using only the top 15 rotamers at each position took 25 s and produced a solution with an energy which was in error by less than 1%. This energy was used as the initial bound for the remaining calculations on the system.

To illustrate the potential for combining DEE and B&T methods by way of DEE preprocessing, we selected a problem computable by either algorithm to enable us to perform quantitative comparisons. In practice, however, the technique is applied to problems that are not currently computable in reasonable computer time by either algorithm, for which the benefit is obviously much greater. Figure 2 illustrates the total calculation times partitioned into DEE and B&T times for optimization of the difficult benchmark consisting of core and boundary residues. The calculations differ in the amount of time allotted to DEE reduction before completion with the B&T algorithm. At the best timing, the combined algorithms complete the optimization eight times faster than DEE alone. Moreover, we have observed that, in practice, the B&T method is generally effective at completing large problems that DEE can reduce to as high as 10^{30} remaining sequences.

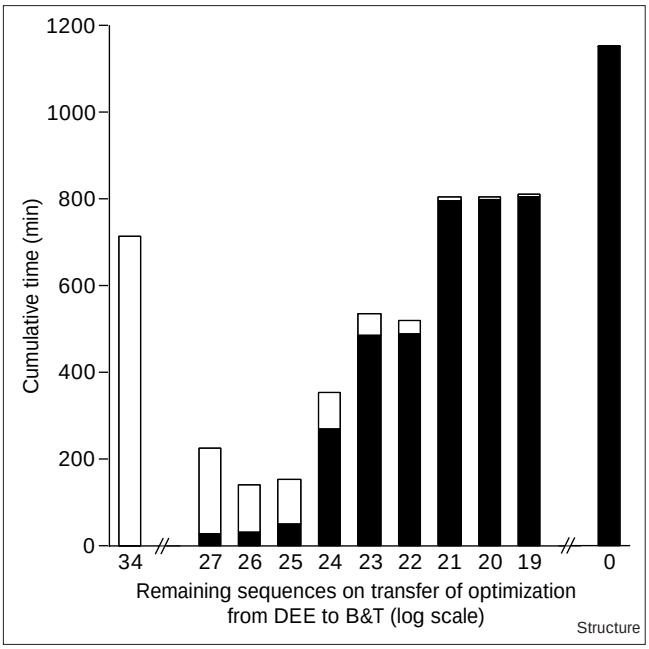
Discussion

We have described a deterministic search method for rotamer optimization and have demonstrated that for some cases it is as fast as the current standard algorithm for protein design, and for other cases it is much faster. The success of the B&T method rests on the construction of a novel pairwise bounding expression, which is used both to perform termination and to supply energetic information with which to determine the search order. Although the algorithm is tailored to protein systems, it can be generalized to any problem that can be similarly described.

Although the B&T algorithm is quite effective when used alone, it is perhaps more important that it increases the problem size limit of DEE calculations by providing an efficient way to complete optimizations for which elimination criteria have become less effective at removing rotamers. This makes it possible to perform optimizations on larger proteins and on systems with large numbers of interacting residues.

The size limit may be raised even higher once the limitations of the approximate form of the algorithm become better understood. For the benchmark cases, the approximate algorithm found the GMEC solutions up to a thousand times faster than either of the exact methods. Even the DEE implementation to which the B&T method is compared incorporates some conservative approximations in the form of high-energy threshold rejection (HETR) criteria [3]. Analogous techniques may provide a way to construct a faster, approximate B&T algorithm with clearly defined accuracy. Along the

Figure 2



Optimization times resulting from the combination of the B&T (white bars) and DEE (solid bars) algorithms. The bars on the extreme left and right of the figure are the times for lone B&T and DEE optimization, respectively. The remaining bars are the cumulative B&T and DEE optimization times when the two algorithms are used in succession. The sudden jumps in DEE times arise from lengthy Goldstein doubles calculations.

same line of reasoning, truncation based on bounding energies might be an effective replacement for HETR cutoffs in DEE.

There is also room for improvement in the heuristic for determining search order. Heuristics that are even more effective may exist that make use of structural information in addition to energetics and size considerations.

In addition, we are currently exploring features of the B&T algorithm that are common to all backtrack searches. First, it is possible to exhaustively sample the amino acid and rotamer sequence space near the GMEC. This is accomplished by modifying the algorithm so that it refrains from lowering the initial minimum energy upon finding low-energy combinations [24]. The result is a full enumeration of all sequences with energies below the specified initial minimum energy, provided that this energy is close enough to the GMEC energy that the calculation remains tractable.

Also, it is straightforward to adapt backtrack algorithms for parallel computation by dispatching branches to different computational nodes. We observe a scaling efficiency between 60 and 80%, depending on the type of problem. Another advantage of the tree representation is that it makes it possible to estimate how much time the optimization will require. This is accomplished using a well-known tree estimation technique [25] in which statistics are compiled for random sample trajectories through the tree. This has helped us to predict when it is best to transfer DEE problems to B&T for completion.

In practice, we believe that the best way to use the B&T method is to first attempt to optimize a problem using DEE. Upon observing that DEE begins to produce very few eliminations or dead-ending pairs, the state information should be transferred to an approximate form of the B&T algorithm. Using the energy from this calculation as the initial upper bound, the approximate algorithm may be repeated again with successively more conservative truncations. The final energy should then be used as the initial bound for the exact B&T calculation.

Biological implications

Protein design and protein homology calculations typically use combinatorial optimization algorithms to compute the optimal placement of amino acid sidechain rotamers on protein backbones. The capabilities of exhaustive search algorithms are currently limited by protein size and energy landscapes. The Branch-and-Terminate variation of the Branch-and-Bound search algorithm described here provides a way to optimize these problems, both alone and used in conjunction with well-established algorithms based on the dead end elimination theorem.

Materials and methods

Benchmark cases

We tested the generality of the algorithm by applying it to a suite of optimization problems representative of different protein structural classes. Rotamers were selected from a backbone-dependent library [26]. To test α-helical surface positions, the 12 residues occupying the (b), (c), and (f) locations in the heptad repeat of one helix of the coiled-cold GCN4-p1 dimer [27] were optimized from the set of rotamers corresponding to hydrophilic amino acids (A, D, E, H, K, N, Q, R, S and T). There were 9.1×10^{22} rotameric combinations.

The β1 domain of streptococcal protein G [28] was used for the remaining cases. As a representative of core and boundary optimization problems, a subset of positions determined to be in the core and boundary according to our residue classification scheme (positions 3, 5, 7, 12, 23, 25, 26, 30, 34, 43, 45, 52 and 54) were optimized from the 3.4×10^{25} combinations of hydrophobic rotamers (amino acids A, F, I, L, M, V, W and Y). For β-sheet surfaces, a subset of the β-sheet surface residues (positions 4, 6, 15, 17, 42, 44, 53 and 55) were optimized from the 4.9×10^{17} combinations of hydrophilic rotamers.

To represent problems consisting of a mixture of different structural types, including turns, we also included the optimization of the residues containing any atoms within 10 Å of the sidechain atoms of Val21. Of these 14, the core residues (positions 3, 20 and 36) were allowed to have any of the hydrophobic identities, the surface residues (positions 2, 19, 21, 22 and 24) had hydrophilic identities, and the remaining boundary residues (positions 1, 18, 23, 25, 27 and 29) were selected from a group of hydrophilic and hydrophobic residues, excluding methionine (amino acids A, D, E, F, H, I, K, L, N, Q, R, S, T, V, W and Y). There were 1.3×10^{29} possible rotameric combinations.

The most difficult benchmark consisted of all 18 nonglycine core and boundary residues [2]. The core residues (positions 3, 5, 7, 20, 26, 30, 34, 39, 52 and 54) were selected from the set of hydrophobic amino acids and the boundary residues (positions 1, 12, 23, 33, 37, 45, 50 and 56) were selected from the composite list of hydrophilic and hydrophobic residues. There were 1.9×10^{34} possible rotameric combinations.

Energy expression

We employ an energy expression that consists of van der Waals, electrostatic and solvation terms. For van der Waals, a Lennard–Jones 6–12 potential was used with radii scaled by a factor of 0.9 [29]. Electrostatics were computed using a distance-dependent dielectric and a hybridization-dependent hydrogen bonding term [30]. Solvation effects were approximated from hydrophobic surface area burial [31]. Atom radii and hydrogen bond well depths were based on the DREIDING force field [32].

Calculation

For reference, calculation times were recorded using a fully optimized DEE algorithm incorporating HETR [3], and magic bullets and other doubles optimizations [14]. Calculations were also performed using an enhanced B&B implementation that employed the efficient bounding criteria and termination preprocessing.

For the first three benchmark cases, all calculations were performed using an initial upper bound of 0.0 kcal/mol, as our energy expression typically results in optimal sequences with negative energies. For the remaining two cases, initial bounds were obtained by first running the approximate version of the algorithm, in which the rotamer lists were truncated to the 15 rotamers with the lowest bounding energies at each residue position. These provided initial bounds of –153.0 and –250.0 kcal/mol, respectively.

The generality of the sorting criteria was demonstrated by performing optimizations with values of f in Equation 5 ranging from 0 to 1.

To illustrate the reliability of the approximate form of the algorithm, optimizations were also performed using only the top 30 rotamers at each position as ranked after a single round of termination.

The larger benchmark problem consisting of core and boundary residues was used to demonstrate how DEE and B&T methods can work in concert. The problem was optimized using a DEE algorithm, and upon every reduction of complexity by at least an order of magnitude, the state of the diminished problem was recorded. A B&T algorithm was used to complete the calculation for each reduced state. The calculations were performed using the optimal sorting factor as determined from the previous benchmarks.

For all calculations, the total CPU time was recorded, as well as the portions of that time spent performing termination preprocessing and the actual recursive search. The total number of nodes comprising the final pruned tree was also recorded by tallying the number of nodes remaining after termination at every level of recursion. Calculations were performed on a single R10000 CPU of a Silicon Graphics Origin 2000.

Pairwise bounding expression

This section describes the construction of a stringent expression for a lower bound for a system composed only of one- and two-body interactions in terms of both a partially specified sequence and the set of rotamers available at its unspecified positions.

For a system consisting only of two-body interactions, the total potential energy can be expressed as the sum of energies between all pairs:

$$E_{\text{total}} = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E(i,j) = \sum_{i} \sum_{\substack{j \\ j>i}} E(i,j) \tag{6}$$

In a protein, i and j refer to amino acid positions and $E(i,j)$ is the energy of interaction between amino acids at those positions.

A protein system also consists of single-body interactions. Because each body is an amino acid sidechain at a particular position on the protein backbone, there is an energy contribution both from sidechain interactions with other sidechains as well as interactions with the protein template scaffolding. Both energies of interaction depend on the sidechain position, amino acid identity, and configuration. Thus the total potential energy can be expressed:

$$E_{\text{total}} = \sum_{i} E(i_c, \text{template}) + \sum_{i} \sum_{\substack{j \\ j>i}} E(i_c, j_c) \tag{7}$$

where c is a position-specific index describing a sidechain rotamer of a particular amino acid type and configuration.

For the purposes of deriving an expression for a lower bound, it is desirable to alter the indices to allow redundancy.

$$E_{total} = \sum_{i} E(i_c, \text{template}) + \tfrac{1}{2} \sum_{i} \sum_{\substack{j \\ j \neq i}} E(i_c, j_c) \tag{8}$$

To ensure that the bounding expression satisfies the condition in Equation 3, we use the following inequalities:

$$\min_{r}[E(i_r, \text{template})] \leq E(i_g, \text{template}) \tag{9}$$

and

$$\min_{r}[E(i_r, j_g)] \leq E(i_g, j_g) \tag{10}$$

in which the indices r and s refer to all of the possible rotamers available at each position, and the minimum operator selects the single rotamer that minimizes the subexpression The index g denotes the rotamer found at the specified position in the global minimum combination. A simple expression for the lower bound is therefore obtained by summing minimal interaction energies between positions by discovering minimal rotamer pairs.

$$E^{(0)}_{\text{bound}} = \sum_{i} \min_{r}[E(i_r, \text{template})] + \tfrac{1}{2} \sum_{i} \min_{r} \sum_{\substack{j \\ j \neq i}} \min_{s}[E(i_r, j_s)] \tag{11}$$

The derivation above represents a generic strategy for producing a bounding expression from any total energy expression. For example, more restrictive bounds can be obtained from energy expressions that sum over three- or four-body interactions. However, the computational cost to implement such bounds on a protein system is very high. Fortunately, there are variations of Equation 7 that are equivalent in terms of computational cost yet yield better bounds.

An alternative way to express the total energy of the system is to distribute the template energies into the pair calculation. Given an energy quantity for a pair of rotamers

$$E_{\text{pair}}(i_c, j_c) = \frac{E(i_c, \text{template}) + E(j_c, \text{template})}{2p - 2} + \frac{E(i_c, j_c)}{2} \tag{12}$$

in which p is the number of amino acid positions, the total energy can be expressed thus:

$$E_{\text{total}} = \sum_{i} \sum_{\substack{j \\ j \neq i}} E_{\text{pair}}(i_c, j_c) \tag{13}$$

which, in turn, can be used to produce the following bounding expression

$$E_{\text{bound}}^{(1)} = \sum_i \min_r \sum_{\substack{j \\ j \neq i}} \min_s [E_{\text{pair}}(i_r, j_s)] \quad (14)$$

Because the minima must be evaluated with respect to single-body and pair energies simultaneously, this bounding expression is necessarily greater than or equal to the expression in Equation 11. Therefore, the new bound is more restrictive. The computational requirements for both expressions, however, are of the same order. Each requires n^2p^2 calculations, where n is the average number of available rotamers per position and p is the number of positions.

One can derive a lower bound that is even more restrictive by performing an expansion of Equation 13 before applying the minimization operators. When testing a particular node during traversal of the combinatorial tree, the positions corresponding to nodes above (and including) the current node have uniquely specified rotamers, whereas the remaining, deeper nodes are not yet uniquely specified. The set of all amino acid positions can therefore be decomposed into two subsets, fixed (F) and variable (V). Equation 13 can be rewritten

$$E_{\text{total}} = \sum_{i \in \{F,V\}} \sum_{\substack{j \in \{F,V\} \\ j \neq i}} E_{\text{pair}}(i_c, j_c) \quad (15)$$

Next, we expand the summation:

$$E_{\text{total}} = \sum_{i \in F} \sum_{\substack{j \in F \\ j \neq i}} E_{\text{pair}}(i_c, j_c) + \sum_{i \in F} \sum_{j \in V} E_{\text{pair}}(i_c, j_c) + \sum_{i \in V} \sum_{j \in F} E_{\text{pair}}(i_c, j_c) + \sum_{i \in V} \sum_{\substack{j \in V \\ j \neq i}} E_{\text{pair}}(i_c, j_c) \quad (16)$$

Application of the minimum operators to this expression would yield a bounding expression equivalent to Equation 14. To increase the stringency, we utilize the inequality

$$\min_r \sum_j E_{\text{pair}}(i_r j_s) \geq \sum_j \min_r E_{\text{pair}}(i_r j_s) \quad (17)$$

The middle two terms of Equation 16 differ only in their indices, and they are therefore equivalent to one another. However, there is a difference once the minimum operators are applied, as the rotamers of the fixed subset (F) will restrict the selection of the minimum energy rotamer pair for the minimized third term, but not for the second. Therefore, we reverse the order of the summation for the second term and combine it with the third term to make use of Equation 17 such that the minimum will be as large as possible:

$$E_{\text{total}} = \sum_{i \in F} \sum_{\substack{j \in F \\ j \neq i}} E_{\text{pair}}(i_c, j_c) + 2 \sum_{i \in V} \sum_{j \in F} E_{\text{pair}}(i_c, j_c) + \sum_{i \in V} \sum_{\substack{j \in V \\ j \neq i}} E_{\text{pair}}(i_c, j_c) \quad (18)$$

Now we apply the minimum operators to all sums over positions that are not uniquely specified:

$$E_{\text{bound}}^{(2)} = \sum_{i \in F} \sum_{\substack{j \in F \\ j \neq i}} E_{\text{pair}}(i_r j_s) + 2 \sum_{i \in V} \min_r \sum_{j \in F} E_{\text{pair}}(i_r j_s) + \sum_{i \in V} \min_r \sum_{\substack{j \in V \\ j \neq i}} \min_s E_{\text{pair}}(i_r j_s) \quad (19)$$

To achieve further stringency, we rearrange Equation 18 before applying the minimum operators:

$$E_{\text{total}} = \sum_{i \in F} \sum_{\substack{j \in F \\ j \neq i}} E_{\text{pair}}(i_c j_c) + \sum_{i \in V} \left\{ 2 \sum_{j \in F} E_{\text{pair}}(i_c, j_c) + \sum_{\substack{j \in V \\ j \neq i}} E_{\text{pair}}(i_c, j_c) \right\} \quad (20)$$

from which we obtain

$$E_{\text{bound}}^{(\text{final})} = \sum_{i \in F} \sum_{\substack{j \in F \\ j \neq i}} E_{\text{pair}}(i_r, j_s) + \sum_{i \in V} \min_r \left\{ 2 \sum_{j \in F} E_{\text{pair}}(i_r j_s) + \sum_{\substack{j \in V \\ j \neq i}} \min_s [E_{\text{pair}}(i_r, j_s)] \right\} \quad (21)$$

The expression can be generalized to any system consisting only of two-body interactions such that the total energy of the system can be expressed as in Equation 13.

Efficient implementation of the bounding expression

The computational cost of evaluating Equation 21 is proportional to p^2n^2, where p is the number of positions and n is the average number of rotamers at each position. When performing termination, the bound is evaluated for all pn rotamers, so that the total calculation order for a round of termination is p^3n^3.

Termination consists of evaluating the bounding expression for rotamers at all the unspecified positions. Therefore, a position is temporarily considered a member of set F while its rotamers are being evaluated. As the expensive second term of the final summation is dependent only on V, its possible values may be precomputed for all rotamers i_r once per position and placed into a table for lookup during the evaluation of Equation 21.

The cost of performing p^2n^2 calculations for assembling the table for the termination of all p positions scales as p^3n^2. The bounding expression now only requires order pn calculations for each of the pn times it is performed, for an overall order of p^2n^2. The overall calculation time therefore scales approximately as p^3n^2, which is nearly n times faster than the direct implementation. Because n is often as large as 100–200, the speed increase can be drastic.

Acknowledgements

We wish to thank AG Street for invaluable feedback throughout the process of developing the algorithm. We also wish to thank EM Reingold, R Manohar, and N Pierce for helpful discussions. This work was supported by the Howard Hughes Medical Institute (SLM), training grant GM 07616C-19 from the National Institutes of Health (DBG), the Rita Allen Foundation, and the David and Lucile Packard Foundation.

References

1. Dahiyat, B.I. & Mayo, S.L. (1997). *De novo* design: fully automated sequence selection. *Science* 278, 82-87.
2. Malakaukas S. & Mayo, S.L. (1998). Design, structure and stability of a hyperthermophilic protein variant. *Nat. Struct. Biol.* 5, 470-475.
3. De Maeyer, M., Desmet, J. & Lasters, I. (1997). All in one: a highly detailed rotamer library improves both accuracy and speed in the modeling of side-chains by dead-end elimination. *Fold. Des.* 2, 53-56.
4. Janin, J., Wodak, S., Levitt, M. & Maigret, D. (1978). Conformation of amino acid side-chains in proteins. *J. Mol. Biol.* 125, 357-386.
5. Ponder, J. & Richards, F. (1987). Tertiary templates for proteins – use of packing criteria in the enumeration of allowed sequences for different structural classes. *J. Mol. Biol.* 193, 775-791.

6. Lee, C. & Levitt, M. (1991). Accurate prediction of the stability and activity effects of site-directed mutagenesis on a protein core. *Nature* **352**, 448-451.
7. Hellinga, H.W. & Richards, F.M. (1994). Optimal sequence selection in proteins of known structure by simulated evolution. *Proc. Natl Acad. Sci. USA* **91**, 5803-5807.
8. Dahiyat, B.I. & Mayo, S.L. (1994). Protein design automation. *Protein Sci.* **5**, 895-903.
9. Koehl, P. & Delarue, M. (1994). Application of a self-consistent mean-field theory to predict protein side-chain's conformation and estimate their conformational entropy. *J. Mol. Biol.* **239**, 249-275.
10. Lee, C. (1994). Predicting protein mutant energetics by self-consistent ensemble optimization. *J. Mol. Biol.* **236**, 918-939.
11. Desmet, J., De Maeyer, M., Hazes, B. & Lasters, I. (1992). The dead-end elimination theorem and its use in protein side-chain positioning. *Nature* **356**, 539-542.
12. Lasters, I. & Desmet, J. (1993). The fuzzy-end elimination theorem – correctly implementing the side-chain placement algorithm based on the dead-end elimination theorem. *Protein Eng.* **6**, 717-722.
13. Goldstein, R.F. (1994). Efficient rotamer elimination applied to protein side-chains and related spin-glasses. *Biophys. J.* **66**, 1335-1340.
14. Gordon, D.B. & Mayo, S.L. (1998). Radical performance enhancements for combinatorial optimization algorithms based on the dead-end elimination theorem. *J. Comp. Chem.* **19**, 1505-1514.
15. Desmet, J., De Maeyer, M. & Lasters, I. (1994). In *The Protein Folding Problem and Tertiary Structure Prediction.* (Merz Jr., K. & Le Grand, S., eds), pp. 307, Birkhäuser, Boston, MA.
16. Reingold, E.M., Nievergelt, J. & Deo, N. (1977). *Combinatorial Algorithms. Theory and Practice.* Prentice-Hall, New Jersey.
17. Hellinga, H.W. & Richards, F.M. (1991). Construction of new ligand binding sites in proteins of known structure. *J. Mol. Biol.* **222**, 763-785.
18. Lathrop, R.H. & Smith, T.F. (1996). Global optimum protein threading with gapped alignment and empirical pair score functions. *J. Mol. Biol.* **255**, 641-665.
19. Escalier, V., Pothier, J., Soldano, H. & Viari, A. (1998). Pairwise and multiple identification of three-dimensional common substructures in proteins. *J. Comp. Biol.* **5**, 41-56.
20. Wang, C.S.E., Lozano-Perez, T. & Tidor, B. (1998). A systematic algorithm for packing of macromolecular structures with ambiguous distance constraints. *Proteins* **32**, 26-42.
21. Todorov, N.P. & Dean, P.M. (1998). A branch-and-bound method for optimal atom-type assignment in de novo ligand design. *J. Comp. Aid. Mol. Des.* **12**, 335-349.
22. Eyrich, V.A., Standley, D.M., Felts, A.K. & Friesner, R.A. (1999). Protein tertiary structure prediction using a branch and bound algorithm. *Proteins* **35**, 41-57.
23. Samudrala, R. & Moult, J. (1998). A graph-theoretic algorithm for comparative modeling of protein structure. *J. Mol. Biol.* **279**, 287-302.
24. Leach, A.R. & Lemon, A.P. (1998). Exploring the conformational space of protein side chains using dead-end elimination and the A* algorithm. *Proteins* **33**, 227-239.
25. Knuth, D.E. (1975). Estimating the efficiency of backtrack programs. *Math.ematics of Computation* **29**, 121-136.
26. Dunbrack, R.L. & Karplus, M. (1993). Backbone-dependent rotamer library for proteins – application to side-chain prediction. *J. Mol. Biol.* **230**, 543-574.
27. E.K. O'Shea, E.K., Klemm, J.D., Kim, P.S. & Alber, T. (1991). X-ray structure of the GCN4 leucine zipper, a 2-stranded, parallel coiled coil. *Science* **254**, 539-544.
28. Gallagher, T., Alexander, P., Bryan, P. & Gilliland, G.L. (1994). Two crystal structures of the β1 immunoglobulin binding domain of streptococcal protein-G and comparison with NMR. *Biochemistry* **33**, 4721-4728.
29. Dahiyat B.I. & Mayo, S.L. (1997). Probing the role of packing specificity in protein design. *Proc. Natl Acad. Sci. USA* **94**, 10172-10177.
30. Dahiyat, B.I., Gordon, D.B. & Mayo, S.L. (1997). Automated design of the surface position of protein helices. *Protein Sci.* **6**, 1333-1337.
31. Street, A.G. & Mayo, S.L. (1998). Pairwise calculation of protein solvent-accessible surface areas. *Fold. Des.* **3**, 253-258.
32. Mayo, S.L., Olafson, B.D. & Goddard W.A. (1990). DREIDING – a generic force-field for molecular simulations. *J. Phys. Chem.* **94**, 8897-8909.

Because *Structure with Folding & Design* operates a 'Continuous Publication System' for Research Papers, this paper has been published on the internet before being printed (accessed from http://biomednet.com/cbiology/str). For further information, see the explanation on the contents page.